UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/625,406	07/23/2003	G. Lawrence Krablin	TN129	9074

7590 09/25/2007

UNISYS Corporation
Unisys Way, MS/E8-114
Blue Bell, PA 19424-0001

EXAMINER
FRANCIS, MARK P

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
09/25/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	**Application No.** 10/625,406	**Applicant(s)** KRABLIN ET AL.	
	Examiner Mark P. Francis	**Art Unit** 2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *21 June 2007*.

2a)☒ This action is **FINAL**. 2b)□ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-3,14-16 and 27-29* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-3,14-16 and 27-29* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)□ The drawing(s) filed on _____ is/are: a)□ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date *09/07/07*.

4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) □ Notice of Informal Patent Application

6) □ Other: _____.

DETAILED ACTION

1. This action is responsive to the amendment filed June 21, 2007.

2. Per applicants' request, claims 1-3,14-16, and 27-29 remain pending.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

4. Claims 1-39 are rejected under 35 U.S.C. 102(e) as being anticipated by Srivastava. (U.S. Pat 5,539,907)

Independent claims

With respect to claims 1,14, and 27, Srivastava discloses a translator (e.g. See Fig. 3, element 51 Translator and related text) operating on a processor for translating compiled programming code from a first code state to a second code state,9Col 4:12-26, "...A compiler translates the high-level language of the program to object code...") the programming code in the first code state comprising a plurality of basic blocks, (Col 3:55-67, "...the procedures including basic blocks...")each basic block comprising a set of instructions,(Col 3:55-62, "...the basic blocks including instructions...") at least one basic block ending in a dynamic branch,(Col 4:1-10, "...by monitoring conditional branch instructions at the end of basic blocks...") the dynamic branch being a transfer to one of a set of destinations based on a calculation of a destination address, (Col 5:25-35, "...addressing schemes...")the translator: identifying the plurality of basic blocks in the first code state of the programming code; (Col 6:1-15, "...The basic blocks..")

identifying links between the identified basic blocks; (Col 5:35-45, "...converts the program into a linked module...")

constructing a control flow graph I representation (CFG) of the programming code based on the identified basic blocks and identified links, the CFG being

in a preliminary form; (Col 6:35-45, "...create the control graphs...", Col 7:55-64, "...procedure flow graph...")

identifying at least one basic block ending in a dynamic branch; (Col 11:14-20, "...a user instrumentation routine branch...")

exploring, based on the CFG, (Col 6:35-45, "...create the control flow graphs...")all identified basic blocks that lead to the dynamic branch as far back as is necessary to fully determine a set of destination addresses for the dynamic branch, the set of destination addresses defining the set of destinations from the dynamic branch; (Col 6:35-53, "...reveals all possible execution destinations...")

examining the set of destinations to identify a branch table; (Col 6:25-40, "...The jump table...a set of branch tables...")

updating the CFG to reflect the set of destinations and the identified branch table; (Col 6:25-40, "...The jump table...a set of branch tables...")

translating the programming code from the first code state to the second code state based at least in part on the updated CFG. (Col 5:37-50, "...The translator converts the program into a linked module...")

Dependent claims

With respect to claims 2,15, and 28, the rejection of claims 1,14, and 27 are incorporated respectively and further, Srivastava discloses that the exploring step comprises the steps of for each explored basic block, constructing a corresponding code graph / representation (code graph) of the instructions in such basic block; (Col 7:55-67, "...procedure flow graph...")and traversing each code graph to determine the set of destination addresses from the dynamic branch.(Col 6:35-41, "...reveals all possible execution destinations...")

With respect to claims 3,16, and 29, the rejection of claims 2,15, and 28 are incorporated respectively and further, Srivastava discloses that each code graph is a rooted directed acyclic graph having interconnected nodes, (Col 5:9-25, "...a program call graph...") each node being one of an instruction node representing an instruction in the corresponding basic block; (Col 3:55-67, "...the basic blocks including instructions...")

an argument node representing an argument in the corresponding basic block;

an apply node edging to an instruction node and to an argument node and representing the application of such argument node to such instruction -node, the apply node in certain instances also being an argument node edged to by another node; (Col 6:25-40, "...The jump table...a set of branch tables...")

a stack node edging to a pair of argument nodes and acting as an argument node having the pair of argument nodes; (Col 7:55-67, "...procedure flow graph...")

a missing argument node representing a missing argument supplied from a different basic block; (Col 5:37-50, "...The translator converts the program into a linked module...")

and an alias node edged to by a stack node or apply node and edging to an argument remote from such stack node, and representing such remote argument to such stack node. (Col 7:55-67, "...procedure flow graph...")

Response to Arguments

5. Applicant's arguments filed on June 21, 2007 have been fully considered but they are not persuasive. Following is the Examiner's response to Applicants' arguments.

With respect to claims 1,14, and 27, Applicant essentially argues that Srivastava et al. does not anticipate the features of this claim because Srivastava et al. does not teach or suggest translating compiled programming code from a first compiled code state to a second compiled code state.

In response, the Examiner differs Note Col 2:38-45, it is here that Srivastava discloses that each of the source code modules is compiled into a corresponding object code modules(First compiled code) then Srivastava teaches that the object code modules are translated into a single linked code module in the form of a machine independent register transfer language. (Second compiled code state). In addition, the Examiner Notes Col 4:12-21, here Srivastava teaches that a compiler translates the high-level language to object code that is then stored in object modules. The object modules are associated with the corresponding relocation tables and symbol tables. Therefore Srivastava does disclose translating compiled programming code from a first compiled code state to a second compiled code state.

In addition, With respect to claims 1,14, and 27, Applicant essentially argues that Srivastava et al. does not anticipate the features of this claim because Srivastava et al. does not teach or suggest dynamic branches.

The Examiner disagrees, Notes Col 6:29-40, it is here that Srivastava teaches that a source-level case-statement is compiled into object code as an indirect jump to an address from some location in a jump table index by the case index value. Srivastava discloses that the jump table for case statements is stored with addresses of different jump target location that can be partitioned into a set of branch tables of a known size. The branch tables contain all of the addresses of all possible execution destinations that is used to build the control flow graph. Thus, the address of the destination of the case statement object code is not known ahead of time but is calculated during program execution. Therefore, Srivastava does teach dynamic branches.

Also, With respect to claims 1,14, and 27, Applicant essentially argues that Srivastava et al. does not anticipate the features of this claim because Srivastava et al. does not teach or suggest exploring, based on a control flow graph all identified blocks that lead to dynamic to fully determine a set of destination addresses for the dynamic branch.

The Examiner disagrees, Notes Col 6:29-40, it is here that Srivastava teaches that a source-level case-statement is compiled into object code as an indirect jump to an address from some location in a jump table index by the case index value. Srivastava discloses that the jump table for case statements is stored with addresses of different jump target location that can be partitioned into a set of branch tables of a known size. The branch tables contain all of the addresses of all possible execution destinations that are used to build the control flow graph. Thus, the address of the destination of the case statement object code is not known ahead of time but is calculated during program execution. Therefore, Srivastava does teach exploring, based on a control flow graph all identified blocks that lead to dynamic to fully determine a set of destination addresses for the dynamic branch.

Lastly, Applicant argues that Srivastava does not teach or suggest translating the programming code from the first compiled code state to the second compiled code state based at least in part on the updated CFG and that the translation occurs before Srivastava's organizer creates the PCG and PFG control graphs.

In reply, the Examiner differs, Note Col 2:38-45, it is here that Srivastava discloses that each of the source code modules is compiled into a corresponding object code modules(First compiled code) then Srivastava teaches that the object code modules are translated into a single linked code module in the form of a machine independent register transfer language. (Second compiled code state). In addition, the Examiner Notes Col 4:12-21, here Srivastava teaches that a compiler translates the high-level language to object code that is then stored in object modules. The object modules are associated with the corresponding relocation tables and symbol tables. Also, the Examiner Note, Col 5:9-17, it is here that Srivastava teaches that the organizer builds a procedure flow graph in memory that maps the flow of control through the basic blocks and indicates how the procedures are called by each other. Later, in Col 5:37-45, Srivastava teaches that the translator converts the program into a linked module in an intermediate form that is a representation of the register transfer language. Therefore Srivastava does disclose translating the programming code from the first compiled code state to the second compiled code state based at least in part on the updated CFG and the translation occurs after the program call graph.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mark P. Francis whose telephone number is (571)272-7956. The examiner can normally be reached on Mon-Fri 8:00-4:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai T.An can be reached on (571)272-3756. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Mark P. Francis

Patent Examiner

Art Unit 2193

MENG-AL T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100